📑 Real-Time Audio Translator - Setup & Usage Guide

Table of Contents

🔧 Installation

Prerequisites

- Python 3.8 or higher
- pip (Python package manager)
- Git (for cloning the repository)

Step 1: Clone the Repository

```bash
git clone https://github.com/yourusername/real-time-translator.git
cd real-time-translator
```

Step 2: Create Virtual Environment (Recommended)

```bash
# Windows
python -m venv venv
venv\Scripts\activate

# macOS/Linux
python3 -m venv venv
source venv/bin/activate
```

Step 3: Install Dependencies

Option A: Standard Installation

```bash
pip install -r requirements.txt
```

Option B: GPU Support (Faster processing)

```bash
# For NVIDIA GPUs with CUDA 11.8
pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118
pip install -r requirements.txt
```

Step 4: Verify Installation

```bash
python main_entry.py --check-deps
```

🚀 Initial Setup

1. First Launch

```bash
python main_entry.py
```

2. Configure Audio Device

For Windows Users:

1. **Physical Microphone**: Select your microphone from the device list
2. **System Audio**:
   - Install VB-Audio Virtual Cable
   - Set virtual cable as Windows default playback device
   - Select "CABLE Output" in the translator

For macOS Users:

1. **Physical Microphone**: Select from device list
2. **System Audio**:
   - Install BlackHole

- Create Multi-Output Device in Audio MIDI Setup
- Select BlackHole in the translator

For Linux Users:

1. Use PulseAudio loopback:

```
bash

pactl load-module module-loopback
```

2. Select "Monitor" device in the application

3. Test Your Setup

1. Click "🔄 Refresh" to scan devices
2. Select your audio source
3. Click "🧪 Test Device"
4. Play audio or speak - you should see the level meter move
5. Adjust threshold based on the test results

4. Load AI Models

1. Click "🤖 Load AI Models"
2. First-time loading downloads models (~500MB-1GB)
3. Wait for "✅ Models Loaded" confirmation
4. Models are cached for future use

📖 Using the Application

Basic Workflow

1. **Start Translation**
   - Ensure models are loaded
   - Select source language (or use Auto-detect)
   - Select target language
   - Click "▶️ Start Translation"
2. **During Translation**
   - Green level meter shows audio activity

- Original text appears in blue
- Translations appear in green
- Timestamps show when each segment was processed

3. **Stop Translation**
   - Click "⏹ Stop Translation"
   - Session statistics are displayed

Understanding the Interface

Audio Configuration Panel

- **Audio Device**: Source for audio capture
- **Audio Level**: Real-time volume indicator
- **Speech Threshold**: Minimum level to trigger processing
- **Audio Gain**: Amplification factor

Language Configuration Panel

- **Source Language**: Language being spoken
- **Target Language**: Language to translate to
- **AI Model**: Whisper model size (accuracy vs. speed)

Output Display Controls

- **Show Logs**: Toggle system messages
- **Show Timestamp**: Toggle time stamps
- **Show Process Time**: Toggle performance metrics
- **Show Original**: Toggle source text display

Common Use Cases

1. Watching Foreign Videos

1. Set system audio as default playback
2. Select video's language as source
3. Select your language as target
4. Start translation and play video

2. Online Meetings

1. Use virtual audio cable
2. Route meeting audio through cable
3. Set source to "Auto-detect"
4. Start translation before meeting

3. Language Practice

1. Use microphone input
2. Set source to language you're practicing
3. Set target to your native language
4. Speak and see instant feedback

⚙️ Advanced Configuration

Optimizing Performance

For Accuracy

```json
{
  "whisper_model": "medium",
  "energy_threshold": 0.005,
  "min_speech_duration": 0.5,
  "silence_timeout": 1.0
}
```

For Speed

```json
{
  "whisper_model": "tiny",
  "energy_threshold": 0.02,
  "min_speech_duration": 0.3,
  "silence_timeout": 0.5
}
```

Custom Language Models

To add new translation pairs:

1. Edit `constants.py`
2. Add model mapping to `TRANSLATION_MODELS`
3. Example:

```python
("ko", "en"): "Helsinki-NLP/opus-mt-ko-en"
```

Adjusting Audio Settings

For Noisy Environments

- Increase threshold to 0.02-0.05
- Decrease gain to 0.5-0.8
- Use noise suppression software

For Quiet Sources

- Decrease threshold to 0.001-0.005
- Increase gain to 2.0-3.0
- Position closer to audio source

💡 Tips & Best Practices

Audio Quality

1. **Use high-quality audio sources** - Clear audio improves accuracy
2. **Minimize background noise** - Use noise cancellation when possible
3. **Maintain consistent volume** - Avoid sudden volume changes

Performance Optimization

1. **Close unnecessary programs** - Free up CPU/RAM
2. **Use GPU if available** - 3-5x faster processing
3. **Start with smaller models** - Upgrade if needed

Language Tips

1. **Auto-detect works best with clear speech**
2. **Specify language when possible** - Faster and more accurate
3. **Similar languages may need manual selection**

Troubleshooting Quick Fixes

Problem	Solution
No audio detected	Check device selection and permissions
Delayed translations	Use smaller model or upgrade hardware
Wrong language detected	Manually select source language
Poor accuracy	Use larger model, improve audio quality
Application crashes	Check RAM usage, restart application

Keyboard Shortcuts (Future Feature)

- Space: Pause/Resume translation
- Ctrl+C: Copy selected text
- Ctrl+S: Save translation history
- Esc: Stop translation

📊 Performance Expectations

Model Comparison

Model	Speed	Accuracy	RAM Usage	Best For
Tiny	⚡⚡⚡⚡⚡	★★	~1GB	Real-time, clear audio
Base	⚡⚡⚡⚡	★★★	~1.5GB	Balanced performance
Small	⚡⚡⚡	★★★★	~2GB	Better accuracy
Medium	⚡⚡	★★★★★	~5GB	Professional use

System Requirements by Use Case

Use Case	Min RAM	Recommended	GPU Benefit
Personal	4GB	8GB	Optional
Professional	8GB	16GB	Recommended
Streaming	8GB	16GB	Required

🆘 Getting Help

Resources

- **Documentation**: Check project README
- **Issues**: GitHub issue tracker
- **Community**: CS50 Discord

Debug Mode

Run with verbose output:

```
bash

python main_entry.py --debug
```

Logs Location

- Windows: `%APPDATA%\RealTimeTranslator\logs`
- macOS: `~/Library/Logs/RealTimeTranslator`
- Linux: `~/.local/share/RealTimeTranslator/logs`

Happy Translating! 🌎🎉